REVIEW RESOURCES

Lesson 22: Software Acquisition: Fundamentals

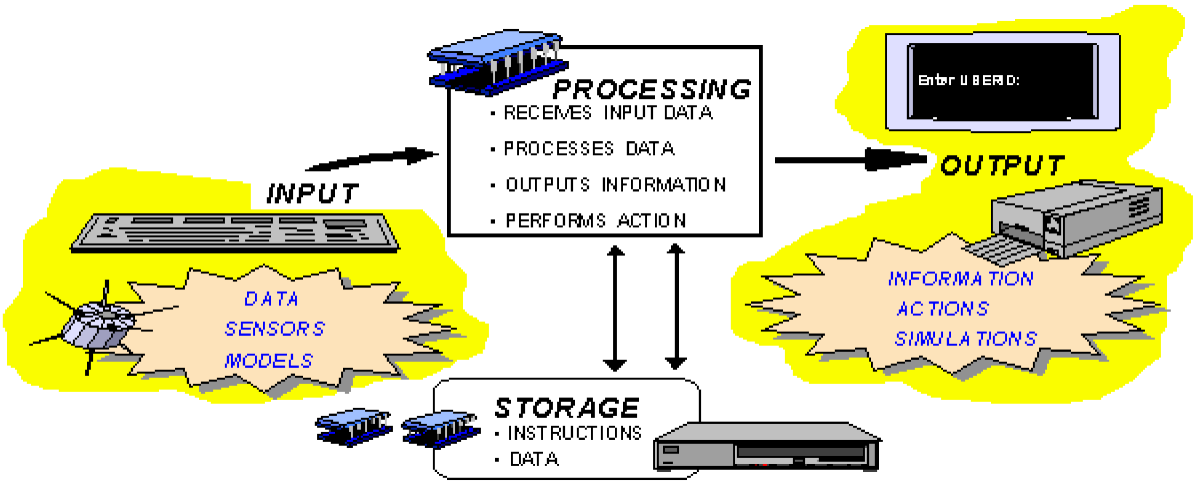
Hardware and Software Basics

Before discussing software acquisition management, it is important to make sure that you have a working knowledge of some basic terms and concepts. This section reviews computer hardware and software basics.

Back to Topics List Back to Top

Hardware Overview

Systems with a high percentage of software functionality include the following four hardware elements: Input, Processing, Storage, and Output. These elements are shown below as part of a PC.



Input Device

A device whose purpose is to allow the user to enter information into a computer system. Common examples include keyboards, mice, and joysticks.

Processing Unit

The processing unit of a computer is the device that interprets and executes instructions. Processing is the vital step between receiving data (input) and producing results (outputs). A central processing unit or CPU is the brain of the computer.

Output Device

An output device makes the processing results (or data) accessible to the user. Common outputs include displays on the computer screen/panels, printed pages, sounds, or files to be stored/sent to another computer.

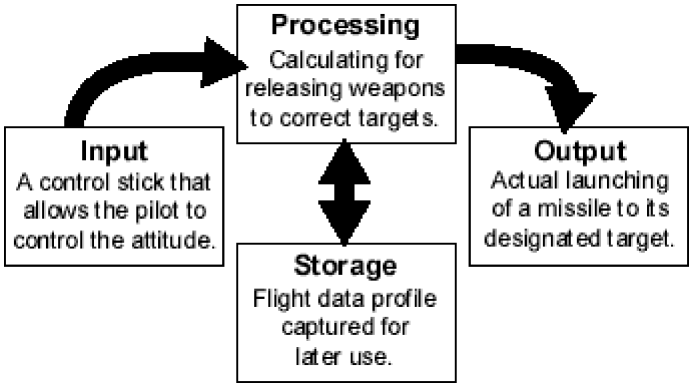
Storage Media

The various types of physical materials on which data are written and stored. Common storage media include floppy disks, hard drives, tape, and CD-ROMs.

Back to Topics List Back to Top

Hardware Elements and Defense Systems

The four hardware elements below appear in many DOD defense systems.



Back to Topics List Back to Top

Software Categories

Software contains the instructions that make the hardware work. Four categories of software are shown below.

- Operating Systems
- Application Software
- Network Software
- Programming Languages

Back to Topics List Back to Top

Operating Systems

Operating systems are the software that controls the allocation and usage of hardware resources such as memory, central processing unit (CPU) time, disk space, and peripheral devices. The operating system is the foundation upon which all other software running in the computer depends.

Popular operating systems include: Windows 95, Windows NT, Mac OS, and UNIX.

Back to Topics List Back to Top

Application Software

Application software performs a specific task, such as word processing, accounting, database management, or inventory control.

Back to Topics List Back to Top

Network Software

Network software permits the connection to or participation in a computer network. This type of software works closely with a computer's operating system.

Note: A network is defined as a group of computers and associated devices that are connected by communications infrastructure (cables, telephone connections, or other links).

Back to Topics List Back to Top

Programming Languages

Programming languages allow programmers to generate the code (or sequence of instructions) that tells the computer system what outputs to produce given certain inputs. All software (operating systems, application software, network software) is written using some type of programming language.

Software code instructs the system what outputs to produce, given certain inputs. This code is expressed in a programming language. Common languages that you may encounter in a defense system are:

- Machine Language
- Assembly Language
- Higher Order Language (HOL)
- Fourth Generation Language (4GL)

Machine Language

Machine Language is:

- The only language the computer speaks.
- Called binary code because the instructions are written either as zero or one. The binary statements control the computer's actions.
- Also referred to as object code.

An example of Machine Language:

Function: Adding two numbers

Machine Language Code:
0001 1100 1010 1111

Assembly Language

Assembly Language is:

- Written in abbreviations or using a mnemonic format. Each mnemonic statement corresponds to one machine language statement.
- Translated into machine language by a computer program called an "assembler."
- Used only for software that must be highly optimized (fast and/or compact).
- Is very hardware dependent.

An example of Assembly Code:

Function: Adding two numbers

Assembly Code:
LD A
ADD B
STR A

Higher Order Language (HOL)

Higher Order Language is:

- A general-purpose programming language that allows programs to be written without having to understand the inner workings of a computer.
- Translated, using a computer program called a "compiler," into a format (object code) executable by a computer.

Hundreds of different HOLs exist. Common HOLs include: Ada, BASIC, C, C++, COBOL, and FORTRAN.

An example of Higher Order Language:

Function: Adding two numbers

Ada Code:
C: = A + B;

Fourth Generation Language (4GL)

A Fourth Generation Language (4GL) is easier for nonprogrammers to use. These languages:

- Are oriented to very specific types of application domain (database access, report generation, etc.)
- Cannot handle as broad a range of applications when compared to HOLs.
- Can be much more productive than HOLs for very specific applications.

Example of a 4GL:

Function: Retrieve Data

Structure Query Language (SQL) Code:
SELECT TABLE 1

Back to Topics List Back to Top

What Is an Architecture?

An architecture can be defined as the structure of components, their relationships, and the principles and guidelines governing their design and evolution over time. In simple terms, an architecture is an in-depth blueprint for constructing and integrating all aspects of a software-intensive system. Following a well thought-out plan:

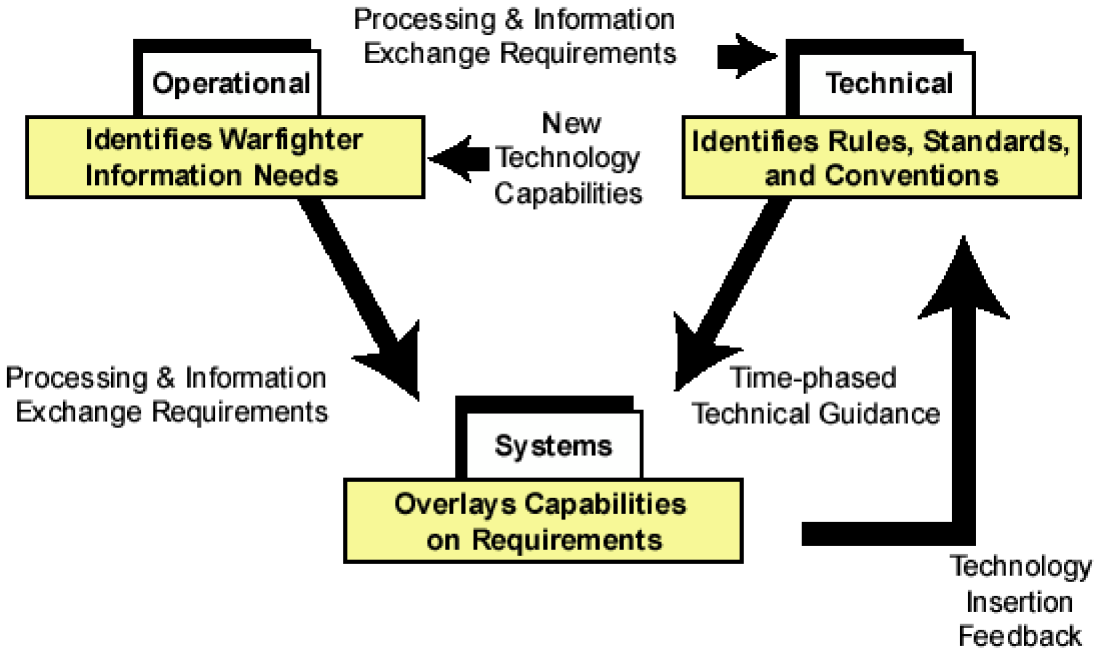
- Makes construction easier.
- Ensures compatibility between different systems or parts.

The Need for Architectures

Frequently, parts of the U.S. forces and allied military forces have to work closely together. Past experiences with joint operations have found that forging integrated international computer-communications networks can be extremely difficult.

Types of Architectures

DOD's Key Architectures



To address interoperability concerns, the DOD has defined an interrelated set of architectures that includes Operational, Systems, and Technical components. These architectures help ensure that U.S. forces and allied military forces can communicate during joint operations.

Operational Architecture

An Operational Architecture is a description of the operational elements, assigned tasks, and information flow required to accomplish or to support the warfighting function.

Technical Architecture

A Technical Architecture is the set of rules, or "building codes" that are used when a system engineer begins to design/specify a system to achieve interoperability.

These rules consist primarily of a common set of standards/protocols to be used for sending and

receiving information, for understanding the information, and for processing that information. This architecture also includes a common human-computer interface and "rules" for protecting the information.

Systems Architecture

A Systems Architecture is a description, including graphics, of systems and interconnections providing for or supporting warfighting functions. This type of architecture:

- Defines the physical connection, location, and identification of the key nodes, circuits, networks, warfighting platforms, etc.
- Specifies system and component performance parameters.
- Shows how multiple systems within a subject area link and interoperate.
- May describe the internal construction or operations of particular systems within the architecture.

Back to Topics List Back to Top

Joint Technical Architecture (JTA)

To foster interoperability, the DOD has adopted the use of the Joint Technical Architecture (JTA). JTA mandates the minimum set of standards and guidelines for the acquisition of all DOD systems that produce, use, or exchange information. These standards apply to:

- Information Processing
- Information Transfer
- Information Modeling
- Human-Computer Interfaces
- Information Systems Security

Back to Topics List Back to Top

What Is an Open System?

An Open System is another important concept that promotes interoperability among DOD systems. An Open System Architecture:

- Uses standards developed and agreed to in an "open" forum by all interested parties to define interfaces, services, and support formats.
- Allows applications to easily be used on different equipment.
- Helps to ensure that systems interact with users in a consistent way.
- Reduces development and training costs.
- Limits impact of hardware changes.
- Fosters the use and integration of commercial-off-the-shelf (COTS) software and hardware products.

Interoperability

Interoperability can be defined in many ways.

The DOD directive on interoperability (DODD 4630.5) defines it as:

> The ability of the systems, units, or forces to provide services to and accept services from other systems, units, or forces, and to use the services so exchanged to enable them to operate effectively together. Interoperability is the condition achieved between systems when information or services are exchanged directly and satisfactorily between

the system and/or their users.

Commercial-Off-The-Shelf (COTS)

Commercial items are those that require no unique Government modifications or maintenance over the life cycle of the product to meet the needs of the procuring agency. An example of a COTS software product is a commercial database management system (DBMS) used as part of a command, control, communications, computers, and intelligence (C4I) system.

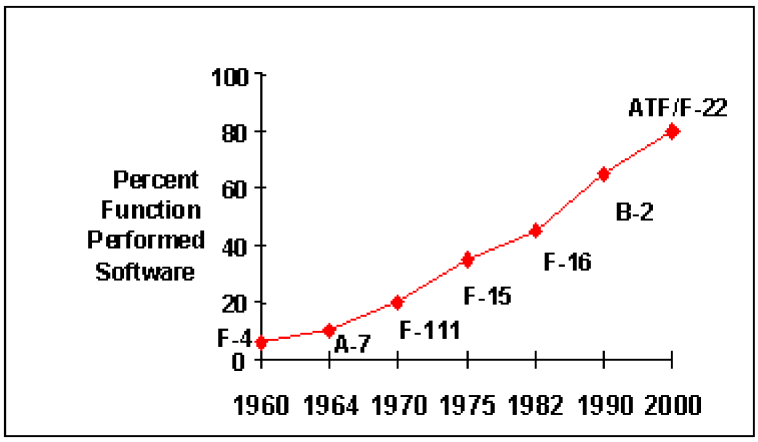
Back to Topics List Back to Top

Importance of Software

Software touches nearly every facet of DOD systems. Some of the more common uses are inventory management and payroll. Less obvious software-intensive systems include aircraft and weapon systems.

Back to Topics List Back to Top

DOD Systems Software Dependencies



The role of software as the most critical part of weapon systems is growing.

For example, the F-22 Advanced Tactical Fighter is currently under development for the Air Force. Eighty percent of the F-22's functionality is dependent on software.

Back to Topics List Back to Top

Software-Intensive System: Definition

A Software-Intensive System is one in which software represents the largest segment in any one or more of the following criteria:

- System development cost
- System development risk
- System functionality
- Development time

Back to Topics List Back to Top

Types of Systems

There are three generic classifications of DOD software-intensive systems:

- Embedded
- Automated Information Systems (AIS)
- Command, Control, Communications, Computers, and Intelligence (C4I).

Embedded Systems

Embedded software is software that is specifically designed into (embedded) or physically integrated into a weapon system. This type of software performs highly specific functions such as weapons firing or navigation. The Patriot Missile or Mission Data Planning System for cruise missiles are examples of embedded systems.

Automated Information Systems (AIS)

Automated Information Systems are defined by the DOD as:

> A combination of computer hardware, software, data, or telecommunications that performs functions such as collecting, processing, transmitting, and displaying information. Excluded are computer resources, both hardware and software that are physically part of, dedicated to, or essential in real time to the mission performance of weapon systems.

Systems in this category may perform administrative functions such as: accounting, payroll, finance, personnel, inventory control, mapping, and equipment maintenance scheduling.

Command, Control, Communications, Computers, and Intelligence (C4I) Systems

C4I systems encompass command, control, communications, computers, and intelligence functions. C4I software is the component of a system that communicates, assimilates, coordinates, analyzes, interprets information, and provides decision support to military commanders.

C4I systems can be used at various levels:

- Strategic C4I systems at the highest levels of command. For example, planning a particular war scenario.
- Tactical C4I systems may be used by units in the field. For example, a field commander might receive target location information to execute a mission. The Global Command and Control System (GCCS) is such a C4I system.

Back to Topics List Back to Top

Common Issues

While each of these three categories of software-intensive systems share many common management and technical issues, there are some distinctions unique to each category. For example:

- Some laws and policies primarily apply only to AIS.
- Embedded software typically has higher development risks and quality requirements.
- C4I systems have stringent interoperability and security needs.

Back to Topics List Back to Top

Acquisition Categories

For acquisition purposes, software-intensive systems are categorized as follows:

Embedded and C4I Systems:

- ACAT I
- ACAT II
- ACAT III

AIS:

- ACAT IA
- ACAT III

Back to Topics List Back to Top

Brooks Act and Warner-Nunn Amendment

For more than 30 years, Federal Government Information Technology (IT) acquisitions, including those of DOD, were mandated by the provisions of the now obsolete 1965 Brooks Act and its 1982 Warner-Nunn amendment.

With the rapid changes in computer technology, the Brooks Act became obsolete. After a number of hearings and studies sponsored by Representative Clinger and Senator Cohen, the Information Technology Management Reform Act (ITMRA) (PL 104-106) was passed in 1996.

The Clinger-Cohen Act of 1996 incorporated the ITMRA and the Federal Acquisition Reform Act (FARA) into a single law.

Back to Topics List Back to Top

Key Thrusts of ITMRA Portion of the Clinger-Cohen Act

The Information Technology Management Reform Act (ITMRA) portion of the Clinger-Cohen Act of 1996 is substantially changing the way agencies acquire Information Technology (IT) within the Federal Government.

The key thrusts of ITMRA are to:

- Require greater accountability for systems improvements achieved through the use of IT.
- Implement performance-based and results-based management.

Back to Topics List Back to Top

Key Provisions of ITMRA (Clinger-Cohen Act)

- Defining terms such as IT and National Security Systems (NSS).
- Repealing the Brooks Act.
- Designating the Office of Management and Budget (OMB) as a focal point to promote IT management within the Federal Government.
- Identifying the Government Accounting Office (GAO) as a single agency protest forum.
- Designating a Chief Information Officer (CIO) for each executive agency.

- Encouraging the use of "Modular Contracting" for categories of systems.

Information Technology

IT is any equipment or interconnected system or subsystem of equipment, that is used in the automatic acquisition, storage, manipulation, management, movement, control, display, switching, interchange, transmission, or reception of data or information. It includes computers, ancillary equipment, software, firmware and similar procedures, telecommunications and communications equipment, services (including support services), and related resources.

National Security Systems (NSS)

NSS are those systems that involve:

- Intelligence activities.
- Cryptologic activities.
- Command and Control of military forces.
- Equipment that is:
  - An integral part of a weapon system, or
  - Critical to the direct fulfillment of military or intelligence missions.

These categories of systems are exempt from some (but not all) of the provisions of the Clinger-Cohen Act.

Back to Topics List Back to Top

Chief Information Officer (CIO)

The designated Chief Information Officer must:

- Have information resources management duties as his or her primary job.
- Monitor and evaluate the performance of information technology programs and advise the agency head on whether to continue, modify, or terminate IT programs.
- Serve as the MDA for ACAT IAM programs.

Back to Topics List Back to Top

DOD Policy Guidelines

The DOD policies for planning computer resources are outlined in DOD 5000.2-R and DODD 5000.1. This section will give an overview of some important guidelines described in these documents.

DOD 5000.2-R

An important document for Software Acquisition Management guidance is the DOD 5000.2-R. This document states:

> "Software shall be managed and engineered using best processes and practices that are known to reduce cost, schedule, and performance risks. It is DOD policy to design and develop software systems based on systems engineering principles."

Software Engineering Policy Guidance

DOD 5000.2-R provides policy guidance for Software Engineering that includes:

- Developing software systems architectures that support open system concepts.
- Identifying and exploiting software reuse opportunities (before beginning new software development).
- Selecting programming languages in context with the systems and software engineering factors.
- Using DOD standard data elements.
- Exploiting the use of COTS products.
- Selecting contractors with:
    - Experience in the software domain (or product line).
    - Successful past performance records.
    - Demonstrable mature development capabilities and processes.
- Using a software measurement process to plan and track software programs.
- Assessing Information Operations risks.
- Ensuring software is Year 2000 (Y2K) compliant.

Architectures That Support Open Systems Concepts

This approach is a business and engineering strategy to use specification and standards that are adopted by industry standard bodies or are de facto standards (set by the marketplace) for selecting system interfaces (functional and physical), products, practices, and tools.

Software Reuse

Software reuse is the process of implementing or updating software systems using existing software assets. An example is reusing the Human-Computer Interface (HCI) software from one C4I system on another system rather than paying for the development of two separate interfaces.

DOD Standard Data Elements

Standardized data results from the process of documenting, reviewing, and approving unique names, definitions, and representations of a data element according to established procedures and conventions. (Source: DOD 8320.1)

Note: A data element is a named identifier of each of the entities and their attributes that are represented in a database.

DODD 5000.1

This directive provides additional guidance on software acquisition management by stating:

> "It is critical that software developers have a successful past performance record, experience in the software domain or product line—a mature software development process and evidence of use and adequate training in software methodologies, tools, and environments."

A variety of frameworks can be used to assess "process maturity."

 Back to Topics List  Back to Top

Software Capability Maturity Model (SW-CMM)

One commonly-used framework for assessing "process maturity" was developed by the Software Engineering Institute (SEI), a DOD agency. The SEI framework is called the Software Capability Maturity Model (SW-CMM). The SW-CMM requires certain key practices to be in place for developers to be assessed at a certain level of process maturity.

There are five levels to the SW-CMM:

- Initial
- Repeatable
- Defined
- Managed
- Optimizing

Initial Level

Organizations at the Initial Level are characterized by:

- Chaotic and ad hoc development.
- The lack of defined processes.
- Relying on heroics (the one "hot-shot" programmer staying up all night) to complete the project.

The majority of software developers are at a lower level.

Repeatable Level

Organizations reach the Repeatable Level when:

- Basic program management processes are established.
- They can repeat earlier success achieved on similar types of projects.

Defined Level

Organizations reach the Defined Level when:

- The processes for both engineering and management activities are documented, standardized, and integrated into a standard process for the organization.
- The organization can tailor standards for a particular project.

Managed Level

Organizations reach the Managed Level when:

- They are able to collect detailed measures of the software process and quality.
- The process and product are understood and controlled.

Optimizing Level

Organizations reach the Optimizing Level when they measure their processes and use these measurements to continuously improve their processes.

Very few software developers are rated at this high level.

Back to Topics List Back to Top